

Unsupervised Detection and Tracking of Arbitrary Objects with Dependent Dirichlet Process Mixtures

Willie Neiswanger* and Frank Wood†

Abstract This paper proposes a technique for the unsupervised detection and tracking of arbitrary objects in videos. It is intended to reduce the need for detection and localization methods tailored to specific object types and serve as a general framework applicable to videos with varied objects, backgrounds, and image qualities. The technique uses a dependent Dirichlet process mixture (DDPM) known as the Generalized Polya Urn (GPUDDPM) to model image pixel data that can be easily and efficiently extracted from the regions in a video that represent objects. This paper describes a specific implementation of the model using spatial and color pixel data extracted via frame differencing and gives two algorithms for performing inference in the model to accomplish detection and tracking. This technique is demonstrated on multiple synthetic and benchmark video datasets that illustrate its ability to, without modification, detect and track objects with diverse physical characteristics moving over non-uniform backgrounds and through occlusion.

1 Introduction

We define unsupervised detection and tracking of arbitrary objects in videos to be the task of automatically identifying the distinct objects present in a sequence of images and determining the path each object follows over time. Techniques that accomplish this task are useful in many fields that make use of video data, including robotics, video surveillance, time-lapse microscopy, and video summarization. By studying this task, we hope

to help make progress towards general machine vision algorithms that can learn the positions, appearances, and number of objects present in any video scene.

This task can be broken down into three parts: data extraction, localization, and tracking. Data extraction is the act of extracting features from regions of video-frames that constitute objects, localization is the act of finding the positions and/or shapes of distinct objects, and tracking is the act of maintaining the identities of the detected objects over time. This paper introduces a new framework for carrying out these three actions based on a type of dependent Dirichlet process mixture model. This framework provides a foundation for a class of unsupervised algorithms that can detect and track arbitrary objects in a wide range of videos.

Research related to general detection and tracking of objects tends to focus on one of either extraction, localization, or tracking. Integrating all three tasks in a system for multiple arbitrary objects and diverse video types is not often a primary focus. A few attempts at accomplishing the three tasks in a cohesive manner have been studied in recent years [8, 9, 25, 49]. This paper furthers this line of work by providing a model that could give rise to a number of algorithms to detect arbitrary objects in videos—particularly in cases where frame-by-frame segmentation is difficult, video quality is low, and extraction is noisy—and maintain the isolation of distinct objects during tracking and through occlusion.

We begin by describing characteristics of the extracted data (Section 3), and giving the generalized form of the model (Section 4). To implement this model, one must specify a data extraction procedure and distributions for representing objects, which may be chosen to allow for arbitrary object tracking or tailored to a specific object type for a given application. In our implementation, we extract data via a basic frame dif-

*Columbia University, Department of Applied Math and Applied Physics. Tel.: 503-464-6152. E-mail: wdn2101@columbia.edu · †Columbia University, Department of Statistics. Tel.: 212-851-2150. E-mail: fwood@stat.columbia.edu

ferencing procedure and specify distributions useful for representing arbitrary objects (Section 5). We describe inference algorithms for our model and show how the output of these algorithms can be interpreted as object localization and tracking results (Section 6). Our implementation is demonstrated on multiple synthetic and benchmark datasets. Standard performance metric values are computed to quantify results on the benchmark datasets (Section 7). We compare the performance metrics of our method with those yielded by specialized detection and tracking algorithms tailored to specific objects in the benchmark datasets. Our results support our hypothesis that approaches combining simple data extraction and a powerful model can perform detection and tracking of arbitrary objects at a level comparable to state-of-the-art, object specific algorithms.

2 Background

A variety of methods in the fields of image processing, signal processing, and computer vision have been developed to solve aspects of the problem of unsupervised detection and tracking of arbitrary objects. These methods might be placed into a few broad categories: those that aim to distinguish the foreground regions of images from the background [33, 12, 65, 39], segment images into distinct regions to perform localization [35, 23, 56], track an object over a sequence of images (after its position has been specified in an initial image) [52, 43, 36, 16, 50], track multiple objects over a sequence of images (especially when the objects interact or occlude one another) [54, 18, 66, 32, 44, 19], segment a sequence of images into distinct spatiotemporal regions [10, 55, 61], and combine the previous methods in some way to create systems capable of both detecting and tracking specified objects [47, 7, 67, 38, 41], or of discerning which regions of a video constitute distinct, arbitrary objects and tracking these [8, 9, 25, 49, 48].

Methods that discern between the foreground and background regions of a video allow for data to be extracted from the areas in each frame where objects reside. Frame differencing and background subtraction are two such methods. Both record locations that exhibit motion relative to the background. Often, background subtraction refers to methods that compare an image containing targets with an image of the background only or with some model of the background that is learned as the video progresses [51], while frame differencing refers to methods that compare pairs of consecutive images in a video [64]. Frame differencing has been used as the sole extraction method for object localization or tracking schemes with success [49, 3, 14],

and also as a secondary data extraction method to help improve the accuracy of object tracking [50].

A great deal of research has focused on developing algorithms to track multiple objects simultaneously. There has been a particular emphasis on developing ways to deal with problems such as object occlusion (where one object blocks another from the view of a video camera) [54, 18, 66], complex object interactions [38, 44, 19], objects with similar appearances [42, 36], variable (and potentially high) numbers of objects [53], and objects that enter and exit a field of view at different times [58, 46]. Multiple independent single-object trackers running simultaneously have been shown to be ineffective, as they will tend to coalesce and track the same object [38]. To remedy this problem, methods have incorporated probabilistic principles for maintaining isolation of object trackers [42]. An approach to this problem involving the use of a nonparametric mixture model has also found success in maintaining isolation of distinct objects [60].

Over the past decade, there have been attempts to provide general algorithms for the fully unsupervised detection and tracking of arbitrary objects in videos. Blob tracking, a basic method for carrying out this goal, has found success in videos where objects are easily isolated from the background and where localization and segmentation of distinct objects is possible [26, 34]. Blob tracking methods, however, run into problems when faced with videos where detection is difficult, object appearance or orientation varies heavily, and there exists object occlusion [57]. To improve the accuracy of these methods, techniques have been developed for performing extraction and segmentation in a joint manner, incorporating statistical methods for maintaining hypotheses of different numbers of detected objects, and introducing some of the multi-object tracking methods described previously to track distinct blobs after they have been segmented [15, 34]. Another family of methods related to the task of unsupervised detection and tracking of video objects goes under the heading of video segmentation algorithms—these methods extend single-frame image segmentation to maintain coherence of image segments over time, and have had some success when used for the explicit purpose of detecting and tracking foreground objects in videos [10, 55, 61]. Other attempts to perform unsupervised detection and tracking include methods for clustering short sequences of positions extracted by detecting the motion of objects [8, 9], which aim to return full-length distinct object tracks, and graph based methods that carry out a similar task using spectral clustering [25]. Another approach uses a Gaussian mixture model to cluster data extracted from moving objects [49]; this method also

develops heuristics for the initialization and elimination of new object tracks.

Nearly all high accuracy object detection and tracking methods are those tailored for specific object types. These methods rely on detection criteria that exploit knowledge about the appearance or behavior of the objects in a video. Some of these methods make use of state-of-the-art detectors designed to locate the specified objects of interest. In contrast, the method described in this paper is designed to track arbitrary objects without using any explicit detection criteria, and serve as a general strategy that can be used, without modification, to perform accurate detection and tracking of diverse objects in a wide range of videos. The technique we introduce falls into the category of clustering-based arbitrary object detection and tracking methods. Differing from previous work, we use a type of time dependent Bayesian nonparametric mixture model, and show how it can be applied to a variety of easily extracted data to perform detection and tracking. This method begins by performing a simple data extraction procedure that yields noisy data. Our model of this data serves as a general framework for which we can choose a variety of object appearance distributions and inference algorithms; each choice provides a new method for unsupervised detection and tracking of arbitrary objects in videos.

3 Data Extraction

We desire a data extraction procedure that yields observations of the form

$$\mathbf{x} = (\mathbf{x}^s, \mathbf{x}^c, t) = (x^{s_1}, x^{s_2}, x^{c_1}, \dots, x^{c_V}, t) \quad (1)$$

where each \mathbf{x} corresponds to a point within an image region where an object (or foreground element) is believed to reside, $\mathbf{x}^s \in \mathbb{R}^2$ denotes the spatial location of this point, $\mathbf{x}^c \in S_1 \times \dots \times S_V$ denotes some collection of local image features in the vicinity of this point, and $t \in \{1, \dots, T\}$ denotes the time index.

We'd like to use an extraction procedure that is as unsophisticated as possible. Consequently, we use frame differencing. This procedure locates the pixels in image regions that undergo change. Specifically, at each frame the pixels that differ from the previous frame beyond some threshold are recorded. Here, each pixel corresponds to an observation \mathbf{x} . Frame differencing is simple, computationally inexpensive, and able to be applied to a wide range of static, single-camera videos (the videos used in experiments are stationary; moving-camera videos require data extraction methods useful

for non-stationary video [12, 65]). Examples of pixel location data extracted via frame differencing are shown in Figure 1(a-j).

We also extract features \mathbf{x}^c that capture image information in the vicinity of each extracted pixel. Examples of possible features include color distributions, pixel intensity values, feature point (such as corner, shape, or edge) locations or spatial characteristics, and texture representations. In principle, we can extract any image features that may be used to characterize the appearance of objects. In our implementation, we choose to extract only color information in the vicinity of each pixel. Incorporating color features allows our method to infer a distribution over color for each detected object; this improves its ability to distinguish between adjacent objects and track objects through occlusion. To add this information, we let \mathbf{x}^c represent a V dimensional discrete distribution over some aspect of color (such as the hue) in the pixel's immediate vicinity. Details on this vector and how it is computed are given in Section 7.2. We refer to the components of this vector as the "color counts" of the pixel.

4 Model Framework

We use a type of dependent Dirichlet process mixture (DDPM) model known as the Generalized Polya Urn dependent Dirichlet process mixture (GPUDDPM). We define a general form of this model in Section 4.1, and specify the distributions used in our implementation of this model in Section 5. We also define a secondary form of this model in Section 6.1, which is used in one of the two inference algorithms. We provide a brief introduction to mixture models and the Dirichlet process in Appendix A.

Dirichlet process mixture (DPM) models (Section A.4) fall under the heading of Bayesian nonparametric models. These models have been widely used in the past decade to perform nonparametric density estimation and cluster analysis. Data extracted from videos comprises spatiotemporal clusters, each corresponding to a distinct object. Consequently, we are interested in using a class of models known as dependent Dirichlet process mixture (DDPM) models (Section A.5), which are particularly useful for estimating the number of latent classes (clusters) in time dependent data.

Since objects can enter and exit a scene, the number of clusters present throughout the video may not be constant (i.e., clusters may be created, or be "born", and may disappear, or "die", at intermediate time steps). To cluster data with these properties, we choose to use a DDPM known as the Generalized Polya Urn dependent Dirichlet process mixture [11]. This model may be

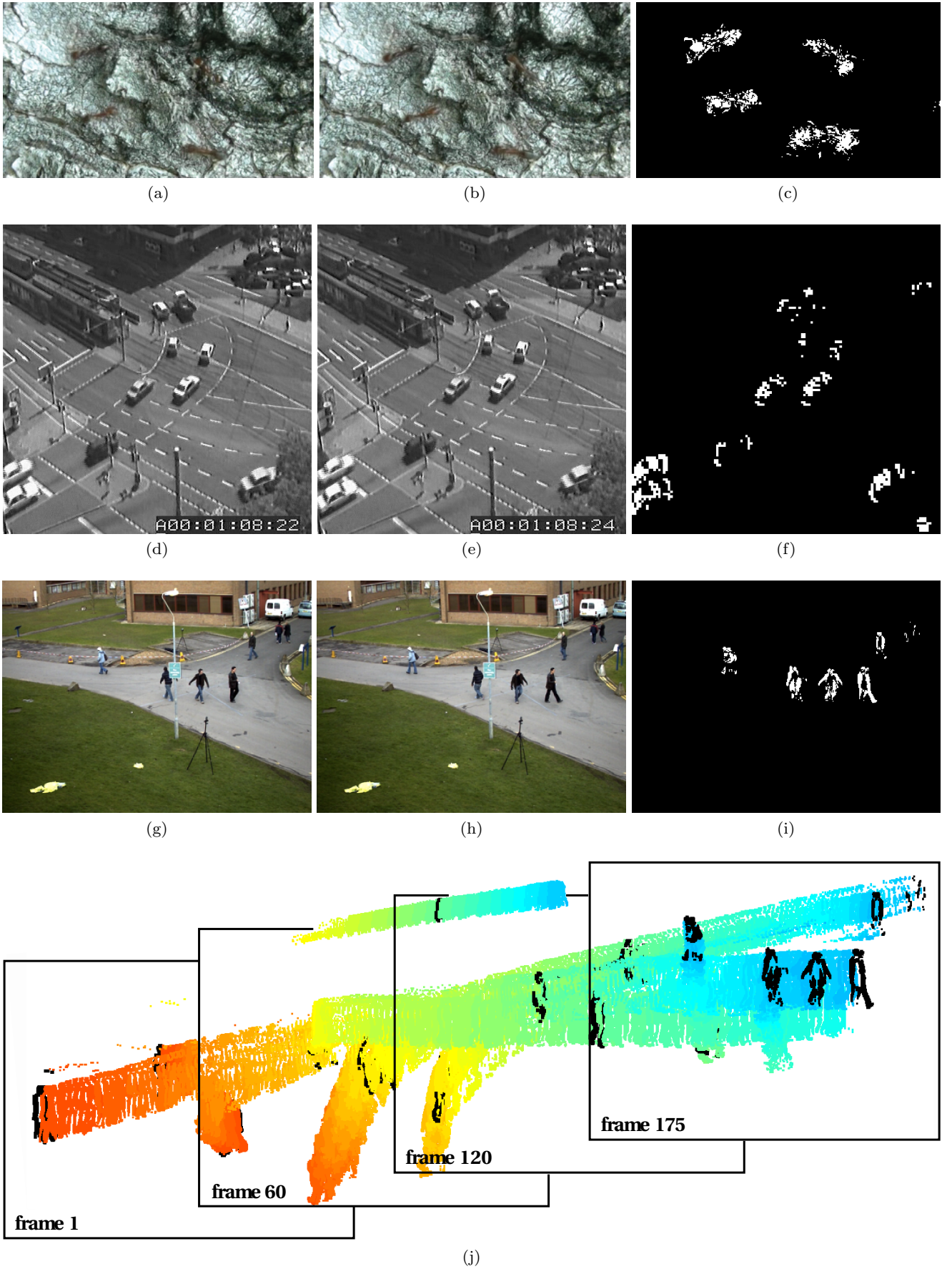


Fig. 1: Three pairs of consecutive frames and the results produced by taking the pixel-wise frame difference between each pair (a - i). The final image shows the results of frame differencing over a sequence of images (from the PETS2009/2010 dataset).

viewed intuitively as a sequence of DPMS, where there exist dependencies between the parameters and number of clusters in adjacent time steps. Like the DPM, the GPUDDPM allows for a distribution over the number of clusters within a dataset—which, in this work, corresponds to the number of objects in a video—to be inferred.

4.1 Generalized Polya Urn Dependent Dirichlet Process Mixture Model

In a GPUDDPM, each observation $\mathbf{x}_{i,t}$ is associated with an assignment variable $c_{i,t}$ that represents its assignment to a cluster $\theta_{k,t}$. The sizes of clusters in the GPUDDPM increase when observations are assigned to them, and decrease at later time points when these observations become “unassigned”. We define a distribution over the size of cluster k at time t ($m_{k,t}$), conditioned on the cluster’s previous size ($m_{k,t-1}$), the assignments at time t ($c_{1:N_t,t}$), and a deletion parameter (ρ),

$$D(m_{k,t}|m_{k,t-1}, c_{1:N_t,t}, \rho) := \text{Binomial}(m_{k,t-1} - m_{k,t} + \sum_{i=1}^{N_t} \mathbb{I}(c_{i,t} = k) | m_{k,t-1}, \rho) \quad (2)$$

$\forall k \in \{1, \dots, K_t\}$, where K_t is the number of clusters at time t and $\mathbb{I}(c_{i,t} = k)$ is an indicator function whose value is 1 if $c_{i,t} = k$ and 0 otherwise.

We also define a distribution over the assignment of observation i at time t ($c_{i,t}$), conditioned on the sizes of all clusters at time t ($m_{1:K_t,t}$) and a concentration parameter (α),

$$C(c_{i,t} = k | m_{1:K_t,t}, \alpha) := \begin{cases} \frac{m_{k,t}}{\sum_{k=1}^{K_t} m_{k,t} + \alpha}, & \text{if } k \in \{1, \dots, K_t\} \\ \frac{\alpha}{\sum_{k=1}^{K_t} m_{k,t} + \alpha}, & \text{if } k = K_t + 1 \end{cases} \quad (3)$$

$\forall i \in \{1, \dots, N_t\}$, where there exists K_t clusters at time t and we give a newly created cluster the index K_{t+1} .

Distributions (2) and (3) together comprise what is referred to as the “Generalized Polya Urn” [11]. We can now define the GPUDDPM generatively as

$$\begin{aligned} m_{k,t} | m_{k,t-1}, c_{1:N_t,t}, \rho &\sim D(m_{k,t-1}, c_{1:N_t,t}, \rho) \\ \theta_{k,t} | \theta_{k,t-1} &\sim \begin{cases} P(\theta_{k,t} | \theta_{k,t-1}) & \text{if } k \leq K_t \\ \mathbb{G}_0 & \text{if } k = K_{t+1} \end{cases} \\ c_{i,t} | m_{1:K_t,t}, \alpha &\sim C(m_{1:K_t,t}, \alpha) \\ \mathbf{x}_{i,t} | c_{i,t}, \theta_{1:K_t,t} &\sim F(\theta_{c_{i,t},t}) \end{aligned} \quad (4)$$

\forall times $t \in \{1, \dots, T\}$ and each cluster $k \in \{1, \dots, K_t\}$ at time t , where we choose application-specific distributions for F , \mathbb{G}_0 , and $P(\theta_{k,t} | \theta_{k,t-1})$ in Section 5. The graphical model associated with this formulation of the GPUDDPM is shown in Figure 2.

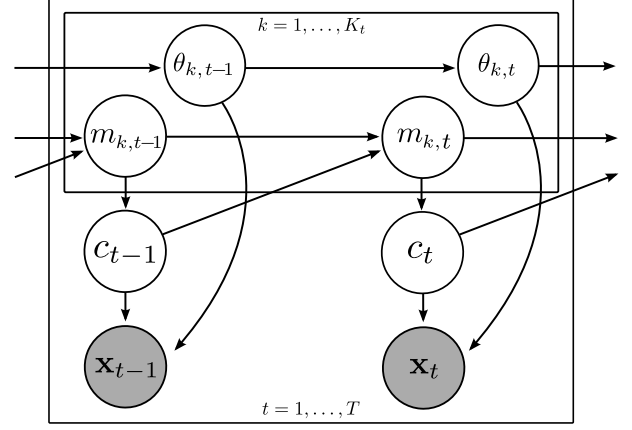


Fig. 2: Graphical Model of the Generalized Polya Urn dependent Dirichlet process mixture. The observations at time t , $\mathbf{x}_{1:N_t,t}$, and their associated assignments $c_{1:N_t,t}$ are denoted respectively as \mathbf{x}_t and c_t (likewise for those at time $t - 1$).

5 Model Specifics

Sections 5.1-5.3 detail the object representation distributions chosen to fully specify our implementation of the GPUDDPM. This specification is used for all experiments in Section 7. The distributions F , \mathbb{G}_0 , and $P(\theta_{k,t} | \theta_{k,t-1})$ represent object appearance, the appearance prior, and object movement, respectively. Our specification is kept general to allow for wide applicability, though one could choose to incorporate known object appearance or motion information for a specific tracking application in future studies.

5.1 Object Appearance and Mixture Component, F

At a given time t , we model each observation $\mathbf{x} \in \mathbf{X}$ as a draw from the product of a multivariate normal and multinomial distribution

$$F(\mathbf{x} | \theta) = \mathcal{N}(\mathbf{x}^s | \boldsymbol{\mu}, \Sigma) \mathcal{Mn}(\mathbf{x}^c | \mathbf{p}) \quad (5)$$

where $\theta = \{\boldsymbol{\mu}, \Sigma, \mathbf{p}\}$ denotes the parameters of a cluster at time t , with mean $\boldsymbol{\mu} \in \mathbb{R}^2$, covariance matrix $\Sigma \in \mathbb{R}^{2 \times 2}$, and discrete probability vector $\mathbf{p} = (p_1, \dots, p_V)$ such that $\sum_{i=1}^V p_i = 1$. Additionally, \mathcal{N} denotes the

multivariate normal distribution and \mathcal{Mn} denotes the multinomial distribution. Note that we withhold writing the subscripts specifying the cluster number k and time t in this section when writing them is unnecessary.

The multivariate normal distribution over the spatial features \mathbf{x}^s models the position and spatial extent of an object. It can be thought intuitively to represent the shape of each object as an oval. We model the color features \mathbf{x}^c as draws from a multinomial distribution. Incorporating this distribution into our cluster likelihood allows us to exploit our observation that the pixels associated with distinct objects tend to have similar color count vectors.

5.2 Appearance Prior and Base Distribution, \mathbb{G}_0

\mathbb{G}_0 denotes the base distribution of the DDPM; it also serves as a prior distribution for the parameters $\theta = \{\boldsymbol{\mu}, \Sigma, \mathbf{p}\}$ of the mixture components (i.e. of the object appearance distributions). We use conjugate priors in the base distribution to allow for more efficient computation. Specifically, in our implementation, a normal-inverse-Wishart prior is placed on the multivariate normal parameters $\{\boldsymbol{\mu}, \Sigma\}$, and a Dirichlet prior is placed on the multinomial parameter \mathbf{p} . The prior can therefore be written

$$\mathbb{G}_0(\theta) = \mathcal{NiW}(\boldsymbol{\mu}, \Sigma | \boldsymbol{\mu}_0, \kappa_0, \nu_0, \Lambda_0) \mathcal{Dir}(\mathbf{p} | \mathbf{q}_0) \quad (6)$$

where \mathcal{NiW} denotes the normal-inverse-Wishart distribution, \mathcal{Dir} denotes the Dirichlet distribution, and the prior has the hyperparameters $\boldsymbol{\mu}_0, \kappa_0, \nu_0, \Lambda_0$ and \mathbf{q}_0 .

5.3 Motion Model and Transition Kernel, $P(\theta_t | \theta_{t-1})$

The transition kernel $P(\theta_t | \theta_{t-1})$ represents how we expect tracked objects to move over time. Since our implementation is intended for tracking arbitrary objects, we do not wish to make sophisticated assumptions about object motion. For example, we choose not to incorporate complex objects dynamics, though they are often used with success in certain object-specific tracking tasks, such as people tracking [13, 17]. We assume only that the position of an object at a given time is close to its position at the previous time, and that the position varies in all directions equally between time steps.

The base distribution \mathbb{G}_0 must be the invariant distribution of the transition kernel $P(\theta_t | \theta_{t-1})$ in order for the cluster parameters to remain marginally distributed according to the base distribution, and for the model to be a valid GPUDDPM. In other words, the

transition kernel must satisfy

$$\int \mathbb{G}_0(\theta_{t-1}) P(\theta_t | \theta_{t-1}) d\theta_{t-1} = \mathbb{G}_0(\theta_t) \quad (7)$$

for a given cluster with parameters θ . One way to achieve this is through the use of auxiliary variables. These are a set of M variables $\mathbf{z}_t = (z_{t,1}, \dots, z_{t,M})$ associated with each cluster at each time t that satisfy

$$P(\theta_t | \theta_{t-1}) = \int P(\theta_t | \mathbf{z}_t) P(\mathbf{z}_t | \theta_{t-1}) d\mathbf{z}_t \quad (8)$$

With the addition of these variables, the parameters of a cluster at a given time do not depend directly on their value at the previous time; they are instead dependent through an intermediate sequence of variables. This allows the cluster parameters at each time step to be marginally distributed according to the base distribution \mathbb{G}_0 while maintaining simple time varying behavior.

Each of the auxiliary variables $z_{t,m}$ is drawn from the product of a multivariate normal and multinomial with the associated cluster parameters $\theta_t = \{\boldsymbol{\mu}_t, \Sigma_t, \mathbf{p}_t\}$

$$z_{t,m} | \boldsymbol{\mu}_t, \Sigma_t, \mathbf{p}_t \sim \mathcal{N}(\boldsymbol{\mu}_t, \Sigma_t) \mathcal{Mn}(\mathbf{p}_t) \quad (9)$$

$\forall m \in \{1, \dots, M\}$. To satisfy (8), we specify the dependencies of a given cluster on its associated set of auxiliary variables at each time t by

$$\boldsymbol{\mu}_t, \Sigma_t, \mathbf{p}_t | \mathbf{z}_t \sim \mathcal{NiW}(\boldsymbol{\mu}_M, \kappa_M, \nu_M, \Lambda_M) \mathcal{Dir}(\mathbf{q}_M) \quad (10)$$

where $\boldsymbol{\mu}_M, \kappa_M, \nu_M, \Lambda_M$, and \mathbf{q}_M are

$$\kappa_M = \kappa_0 + M \quad (11)$$

$$\nu_M = \nu_0 + M \quad (12)$$

$$\boldsymbol{\mu}_M = \frac{\kappa_0}{\kappa_0 + M} \boldsymbol{\mu}_0 + \frac{M}{\kappa_0 + M} \bar{\mathbf{z}}^s \quad (13)$$

$$\Lambda_M = \Lambda_0 + S_{\mathbf{z}_t^s} \quad (14)$$

$$\mathbf{q}_M = \mathbf{q}_0 + \sum_{m=1}^M z_{t,m}^c \quad (15)$$

and where M is the number of auxiliary variables, $\{\boldsymbol{\mu}_0, \kappa_0, \nu_0, \Lambda_0\}$ are the \mathcal{NiW} prior parameters, and \mathbf{q}_0 is the \mathcal{Dir} prior parameter. We use \mathbf{z}^s and \mathbf{z}^c to respectively denote the spatial and color features of an auxiliary variable \mathbf{z} , and $\bar{\mathbf{z}}$ and $S_{\mathbf{z}}$ to respectively denote the sample mean and sample covariance for a set $\mathbf{z} = \{z_1, \dots, z_M\}$ (of auxiliary variables, in this case), which we can write as

$$\bar{\mathbf{z}} = \left(\sum_{m=1}^M z_m \right) / M \quad (16)$$

$$S_{\mathbf{z}} = \sum_{m=1}^M (z_m - \bar{\mathbf{z}})(z_m - \bar{\mathbf{z}})^T \quad (17)$$

5.4 Recap of Model Parameters

The multivariate normal-multinomial GPUDDPM for object tracking has a number of parameters, which are used to specify the object appearance prior distribution, transition kernel, and Generalized Polya Urn distributions (C and D). The object appearance prior parameters include:

$\boldsymbol{\mu}_0 \in \mathbb{R}^2$	Mean position prior. In experiments performed in Section 7 the data was recentered to the origin, and this parameter was set to $(0, 0)$.
$\kappa_0 \in \mathbb{R}$	Scale factor of the mean prior.
$\Lambda_0 \in \mathbb{R}^{2 \times 2}$	Shape factor of the covariance prior.
$\nu_0 \in \mathbb{Z}_+ > 1$	Scale factor of the covariance prior.
$\mathbf{q}_0 \in \mathbb{R}_+^V$	Scale factor of the multinomial prior.

The following parameter dictates characteristics of object movement.

$M \in \mathbb{Z}_+$	Number of auxiliary variables. A larger number will produce a smoother object path.
----------------------	---

Additionally, one can tune the model’s tendency to detect new objects and maintain the existence of these objects (both dictated by distributions C and D) with the following parameters.

$\alpha \in \mathbb{R}_+$	The concentration parameter for the Dirichlet process. A higher value will increase the tendency for new objects to be detected.
$\rho \in (0, 1]$	The deletion parameter. A higher value will give objects an increased tendency to die off.

6 Inference

Bayesian inference is used to achieve detection and tracking results. Previously developed inference strategies can be applied to the generative model defined in Section 4.1 and Section 5. We provide details on the two Bayesian inference algorithms implemented in this study. The first is a type of Markov Chain Monte Carlo (MCMC) batch inference, which uses Gibbs sampling to generate samples from the posterior distribution of the model. The second is a type of Sequential Monte Carlo (SMC) inference, also known as a particle filter, which generates samples from the posterior distribution of the model in a sequential manner.

6.1 MCMC: Batch Inference

This section details the MCMC sampler used to perform inference. A secondary formulation of the GPUDDPM, which we refer to as the “Deletion Variable Formulation” (defined in Section 6.1.1), is used here. This formulation is equivalent to the formulation given in Section 4.1, but allows for easier sampling.

6.1.1 Deletion Variable Formulation

Instead of incorporating the cluster size variable $m_{k,t}$ directly in the GPUDDPM model (as it was in the definition given by (4)), we can formulate an equivalent model which makes use of new set of variables called deletion variables. We introduce a deletion variable $d_{i,t}$ for each observation $\mathbf{x}_{i,t}$, which denotes the time at which the observation is removed from its assigned cluster. At each time, cluster sizes $m_{k,t}$ can be reconstructed from all previous assignments and deletion variables by

$$m_{k,t} = \sum_{t'=1}^t \mathbb{I}[(c_{t'} = k) \wedge (t < d_{t'})] \quad (18)$$

where $\mathbb{I}[\cdot]$ is an indicator function that evaluates to 1 if its argument is true, and 0 otherwise. Additionally, we can define the deletion variable $d_{i,t}$ to be $d_{i,t} = t + l_{i,t}$, where $l_{i,t}$ can be thought of as the lifetime of an assignment. From the definition of distribution D (given by (2)), the lifetime can be shown to be distributed geometrically, and can be written as

$$l_{i,t} | \rho \sim \rho(1 - \rho)^{l_{i,t}} \quad (19)$$

where the parameter ρ is the same as that in (2).

We can now define the Deletion Variable Formulation of the GPUDDPM generatively as

$$\begin{aligned} d_{i,t} | \rho &\sim \text{Geo}(\rho) + t + 1 \\ \theta_{k,t} | \theta_{k,t-1} &\sim \begin{cases} P(\theta_{k,t} | \theta_{k,t-1}) & \text{if } k \leq K_t \\ \mathbb{G}_0 & \text{if } k = K_t + 1 \end{cases} \\ c_{i,t} | \mathbf{c}_{1:t-1}, \mathbf{d}_{1:t-1}, \alpha &\sim \text{C}(\mathbf{c}_{1:t-1}, \mathbf{d}_{1:t-1}, \alpha) \\ \mathbf{x}_{i,t} | c_{i,t}, \theta_{c_{i,t},t} &\sim \text{F}(\theta_{c_{i,t},t}) \end{aligned} \quad (20)$$

\forall times $t \in \{1, \dots, T\}$ and clusters $k \in \{1, \dots, K_t\}$ at time t , where $\mathbf{c}_t = c_{1:N_t,t}$, $\mathbf{d}_t = d_{1:N_t,t}$, and the distributions F , \mathbb{G}_0 , and $P(\theta_{k,t} | \theta_{k,t-1})$ are described in Section 5. This formulation of the GPUDDPM is also used by [28] and [11]. The associated graphical model for this formulation is given in Figure 3.

For easier notation, we define $\mathbf{x}_t = \mathbf{x}_{1:N_t,t}$, $\mathbf{c}_t = c_{1:N_t,t}$, $\mathbf{d}_t = d_{1:N_t,t}$, $\boldsymbol{\Theta}_t = \theta_{1:K_t,t}$, and $\mathbf{z}_t = z_{1:K_t,t,1:M}$.

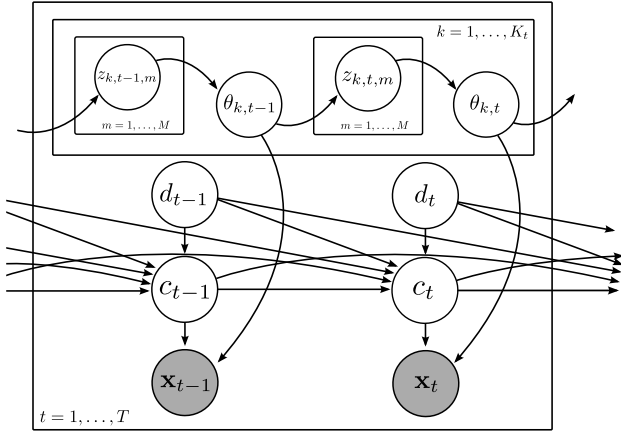


Fig. 3: Graphical model of the Deletion Variable Formulation of the GPUDDPM, showing auxiliary variables.

At each time t , the sampler moves sequentially through the N_t observations, sampling the assignment $c_{i,t}$ and the deletion variable $d_{i,t}$ for each. Afterwards, the cluster parameters for all active clusters at t are sampled, and the M auxiliary variables for all active clusters at t are sampled using Metropolis Hastings (MH). The following sections detail the distributions from which each of these samples is drawn.

6.1.2 Sampling Assignment Variables, $c_{i,t}$

A value proportional to the posterior probability can be computed for each possible value that $c_{i,t}$ may take on. These values allow us to construct a discrete probability distribution from which we can draw samples from the posterior distribution over assignments. The possible values that the $c_{i,t}$ may take on are $k \in \{1, \dots, K'_t + 1\}$, where K'_t denotes the number clusters with a non-zero size at any time $t' \in \{t, \dots, d_{i,t}\}$, and $K'_t + 1$ denotes a “new” cluster. The probability that $x_{i,t}$ is assigned to cluster k , i.e. that $c_{i,t} = k$, given values for all other variables in the model (which we denote as “...”), is given by

$$p(c_{i,t} = k | \dots) \propto \prod_{i'=i}^{N_t} C(c_{i',t} | \mathbf{m}_t, \alpha) \times \prod_{t'=t+1}^{d_{i,t}} \prod_{i'=1}^{N_{t'}} C(c_{i',t'} | \mathbf{m}_{t'}, \alpha) \times \begin{cases} F(\mathbf{x}_{i,t} | \theta_{k,t}) & \text{if } k \leq K'_t \\ \int P(\mathbf{x}_{i,t} | \theta) \mathbb{G}_0(\theta) d\theta & \text{if } k = K'_t + 1 \end{cases} \quad (21)$$

where C is given by (3), and the cluster sizes $\mathbf{m}_{t'}$ are calculated under the assumption that $c_{i,t} = k$. Note

that the above integral has an analytic solution for our specific model

$$\begin{aligned} & \int P(\mathbf{x}_i | \theta) \mathbb{G}_0(\theta) d\theta \\ &= t_{\nu_0-1} \left(\mathbf{x}_i^s \mid \boldsymbol{\mu}_0, \frac{\Lambda_0(\kappa_0 + 1)}{\kappa_0(\nu_0 - 1)} \right) \\ & \times \prod_{v=1}^V \frac{\Gamma(\mathbf{x}_i^c)}{\Gamma(\mathbf{q}_0)} \times \frac{\Gamma(\sum_{j=1}^V \mathbf{q}_0)}{\Gamma(\sum_{j=1}^V \mathbf{x}_i^c)} \end{aligned} \quad (22)$$

where t denotes the multivariate t-distribution, where we follow the three-value parameterization (location parameter, scale parameter, and degrees of freedom) given in [30, 62], and $\{\boldsymbol{\mu}_0, \kappa_0, \Lambda_0, \nu_0, \mathbf{q}_0\}$ are prior parameters.

If a new cluster is sampled as an assignment, the cluster parameters and auxiliary variables for this new cluster must be initialized for all time steps before sampling can proceed. In our implementation, newly sampled clusters were initialized by iteratively sampling forward to time T and backwards to time 1 via the transition kernel.

6.1.3 Sampling Cluster Parameters, $\theta_{k,t}$

The conjugacy of appearance model and transition kernel distributions allow us to easily sample from the posterior distribution over the cluster parameters, which we can write

$$\begin{aligned} P(\theta_{k,t} | \dots) &= P(\mathbf{x}_{i,t} | \theta_{k,t}) P(z_{k,t+1:1:M} | \theta_{k,t}) \\ & \times P(\theta_{k,t} | z_{k,t,1:M}) \\ &= \mathcal{NiW}(\boldsymbol{\mu}_{k,t}, \Sigma_{k,t} | \boldsymbol{\mu}_N, k_N, v_N, \Lambda_N) \\ & \times \mathcal{Dir}(\mathbf{p}_{k,t} | \mathbf{q}_N) \end{aligned} \quad (23)$$

Where the parameters in the above distribution are given when the observations $\mathbf{x}_{1:N_t,t}$, and auxiliary variables $z_{k,t-1:t,1:M}$ for cluster k at time $t-1$ and t , are taken to be the “observations” for the following Bayesian updates

$$\kappa_N = \kappa_0 + N \quad (24)$$

$$\nu_N = \nu_0 + N \quad (25)$$

$$\boldsymbol{\mu}_N = \frac{\kappa_0}{\kappa_0 + N} \boldsymbol{\mu}_0 + \frac{N}{\kappa_0 + N} \bar{\mathbf{x}}^s \quad (26)$$

$$\Lambda_N = \Lambda_0 + \mathbf{S}_{\mathbf{x}^s} \quad (27)$$

$$\mathbf{q}_N = \mathbf{q}_0 + \sum_{i=1}^N \mathbf{x}_i^c \quad (28)$$

where N is the number of observations, $\{\boldsymbol{\mu}_0, \kappa_0, \nu_0, \Lambda_0\}$ are the \mathcal{NiW} prior parameters, \mathbf{q}_0 is the \mathcal{Dir} prior parameter, \mathbf{x}^s and \mathbf{x}^c respectively denote the spatial and color features of the observations, and $\bar{\mathbf{x}}$ and $\mathbf{S}_{\mathbf{x}}$ respectively denote the sample mean and sample covariance for the set of observations \mathbf{x} , defined in (16) and (17).

6.1.4 Sampling Auxiliary Variables, $z_{k,t,m}$

The posterior distribution over each of the auxiliary variables $z_{k,t,m}$ can be written as

$$P(z_{k,t,m} | \dots) \propto P(z_{k,t,m} | \theta_{k,t-1}) P(\theta_{k,t} | z_{k,t,1:M}) \quad (29)$$

We sample a new value for all M auxiliary variables, denoted $z_{k,t,m}^*$, using MH, with the proposal distribution

$$\begin{aligned} z_{k,t,m}^* &\sim P(z_{k,t,m} | \theta_{k,t}) \\ &= \mathcal{N}(z_{k,t,m}^s | \boldsymbol{\mu}_{k,t}, \Sigma_{k,t}) \mathcal{Mn}(z_{k,t,m}^c | \mathbf{p}_{k,t}) \end{aligned} \quad (30)$$

and compute the standard MH acceptance ratio, which in this case simplifies to

$$r_{\text{accept}} = \frac{P(\theta_{k,t-1} | z_{k,t,m}^*)}{P(\theta_{k,t-1} | z_{k,t,m})} \quad (31)$$

6.1.5 Sampling Deletion Variables, $d_{i,t}$

Sampling deletion variables could be performed in a manner similar to how we sample the assignment variables in Section 6.1.2, but this may be computationally expensive due to the large number of possible deletion times. To remedy this, the MH algorithm may again be used to generate samples $d_{i,t}^*$ from the posterior distribution over possible deletion times, where we use the proposal distribution

$$\begin{aligned} l_{i,t} &\sim \text{Geo}(\rho) \\ d_{i,t}^* &= l_{i,t} + t + 1 \end{aligned} \quad (32)$$

where $l_{i,t}$ denotes the geometrically distributed “life-time”, and we accept or reject this sample using the process described in Section 6.1.4.

6.2 SMC: Sequential Inference

This section details a Sequential Monte Carlo (SMC) sampler—also known as a particle filter—used to perform inference. SMC inference operates in the original GPUDDPM formulation (Section 4.1). The algorithm is shown in Algorithm 1. At each time step $t \in \{1, \dots, T\}$, a number of samples referred to as “particles” are generated; each particle consists of a sample from the posterior distribution over the assignment for each observation, $c_{1,t}, \dots, c_{N_t,t}$, parameters for each cluster $\theta_{1,t}, \dots, \theta_{K_t,t}$, and size after deletion for each cluster $m_{1,t}, \dots, m_{K_t,t}$. A set of particles is sampled at each time step from relevant proposal distributions (described in Sections 6.2.1, 6.2.2, and 6.2.3), a weight is computed for each particle, and a new set of particles are sampled

from the set of weighted particles via a resampling process. Within each time step, Gibbs sampling is used to generate the samples associated with each particle.

The sequence of target distributions for the SMC algorithm may be written as

$$\begin{aligned} \pi_t(\mathbf{c}_{1:t}, \boldsymbol{\theta}_{1:t}, \mathbf{m}_{1:t}) &= \pi_{t-1}(\mathbf{c}_{1:t-1}, \boldsymbol{\theta}_{1:t-1}, \mathbf{m}_{1:t-1}) \\ &\times \prod_{i=1}^{N_t} P(c_{i,t} | \mathbf{m}_t, \boldsymbol{\theta}_t, \mathbf{c}_{1:t}, \mathbf{x}_{1:N_t}) \\ &\times \prod_{k=1}^{K_t} \begin{cases} P(\theta_{k,t} | \theta_{k,t-1}) & \text{if } k \leq K_{t-1} \\ \mathbb{G}_0 & \text{if } k > K_{t-1} \end{cases} \\ &\times \prod_{k=1}^{K_t} D(m_{k,t} | m_{k,t-1}, c_{1:N_{t-1},t-1}, \rho) \end{aligned} \quad (33)$$

where $\mathbf{c}_t = c_{1:N_t,t}$, $\boldsymbol{\theta}_t = \theta_{1:K_t,t}$, and $\mathbf{m}_t = m_{1:K_t,t}$, and D is given by (2).

6.2.1 Proposal Distribution for Assignments

The probability of assignments given current cluster sizes, cluster parameters, and the Dirichlet process concentration parameter α can be written as

$$\begin{aligned} P(c_{i,t} | m_{1:K_t,t}, \theta_{1:K_t,t}, \alpha) &\propto C(m_{1:K_t,t}, \alpha) \\ &\times \begin{cases} F(\mathbf{x}_{i,t} | \theta_{c_{i,t},t}) & \text{if } k \leq K_{t-1} \\ \int P(\mathbf{x}_{i,t} | \theta) \mathbb{G}_0(\theta) d\theta & \text{if } k > K_{t-1} \end{cases} \end{aligned} \quad (34)$$

where C is defined in (3), and $\int P(\mathbf{x}_{i,t} | \theta) \mathbb{G}_0(\theta) d\theta$ can be determined analytically, and is given in (22).

6.2.2 Proposal Distribution q_1

The following is a distribution over cluster parameters $\theta_{k,t}$ given a set of N observations $\mathbf{x}_{1:N,t}$. We define q_1 to be

$$q_1(\theta_{k,t} | \mathbf{x}_{1:N,t}) = P(\theta_{k,t} | \mathbf{x}_{1:N,t}) \quad (35)$$

where samples can be drawn from $P(\theta_{k,t} | \mathbf{x}_{1:N,t})$ using the Bayesian updates found in (24), (25), (26), and (28), where the $\mathbf{x}_{1:N,t}$ are taken to be the observations.

6.2.3 Proposal Distribution q_2

The following is a distribution over cluster parameters $\theta_{k,t}$ given a set of N observations $\mathbf{x}_{1:N,t}$ and the cluster parameters at a previous time, $\theta_{k,t-1}$. We define q_2 to be

$$q_2(\theta_{k,t} | \theta_{k,t-1}, \mathbf{x}_{1:N,t}) = P(\theta_{k,t} | \theta_{k,t-1}, \mathbf{x}_{1:N,t}) \quad (36)$$

Algorithm 1 Sequential Monte Carlo Inference for the GPUDDPM

```

1: for  $l = 1 : L$  do
2:    $w_0^{(l)} \leftarrow 1/L$  ▷ initialize weights
3: end for
4:  $K_0^{(l)} \leftarrow 0$  ▷ initialize # of clusters
5: for  $t = 1 : T$  ( $\#$  of frames) do
6:   for  $l = 1 : L$  ( $\#$  of particles) do
7:      $K_t^{(l)} \leftarrow K_{t-1}^{(l)}$ 
8:      $m_{1:K_t^{(l)},t}^{(l)} \leftarrow m_{1:K_{t-1}^{(l)},t-1}^{(l)}$ 
9:     for  $s = 1 : S$  ( $\#$  of Gibbs samples) do
10:      for  $i = 1 : N_t$  ( $\#$  of observations at frame  $t$ ) do
11:        if  $s = 1$  then
12:          Sample  $c_{i,t}^{(l)} \sim P\left(c_{i,t}^{(l)} | m_{1:K_{t-1}^{(l)},t-1}^{(l)}, \theta_{1:K_{t-1}^{(l)},t-1}^{(l)}, \alpha\right)$  ▷ eq. (34)
13:           $m_{c_{i,t}^{(l)},t}^{(l)} \leftarrow m_{c_{i,t}^{(l)},t}^{(l)} + 1$ 
14:        else
15:           $m_{c_{i,t}^{(l)},t}^{(l)} \leftarrow m_{c_{i,t}^{(l)},t}^{(l)} - 1$ 
16:          Sample  $c_{i,t}^{(l)} \sim P\left(c_{i,t}^{(l)} | m_{1:K_t^{(l)},t}^{(l)}, \theta_{1:K_t^{(l)},t}^{(l)}, \alpha\right)$  ▷ eq. (34)
17:           $m_{c_{i,t}^{(l)},t}^{(l)} \leftarrow m_{c_{i,t}^{(l)},t}^{(l)} + 1$ 
18:        end if
19:        if  $c_{i,t}^{(l)} = K_t^{(l)} + 1$  (a new cluster) then
20:          Sample  $\theta_{c_{i,t}^{(l)},t}^{(l)} \sim q_1(\mathbf{x}_{i,t})$  ▷ eq. (35)
21:           $K_t^{(l)} \leftarrow K_t^{(l)} + 1$ 
22:           $m_{K_t^{(l)},t}^{(l)} \leftarrow 1$ 
23:        end if
24:      end for
25:      for  $k = 1 : K_t^{(l)}$  ( $\#$  of clusters at frame  $t$ ) do
26:        if  $k > K_{t-1}^{(l)}$  then
27:          Sample  $\theta_{k,t}^{(l)} \sim q_1(\{\mathbf{x}_{1:N_t,t} = k\})$  ▷ eq. (35)
28:        else if  $k \leq K_{t-1}^{(l)}$  and  $\#\{\mathbf{x}_{1:N_t,t} = k\} > 0$  then
29:          Sample  $\theta_{k,t}^{(l)} \sim q_2(\theta_{k,t-1}^{(l)}, \{\mathbf{x}_{1:N_t,t} = k\})$  ▷ eq. (36)
30:        else if  $m_{k,t}^{(l)} > 0$  then
31:          Sample  $\theta_{k,t}^{(l)} \sim P(\theta_{k,t}^{(l)} | \theta_{k,t-1}^{(l)})$  ▷ eqs. (9) & (10)
32:        end if
33:        if  $s = S$  then
34:          Sample  $m_{k,t+1}^{(l)} \sim D(m_{k,t}^{(l)}, c_{1:N_t,t}^{(l)}, \rho)$  ▷ eq. (2)
35:        end if
36:      end for
37:    end for
38:     $\tilde{w}_t^{(l)} \leftarrow w_{t-1}^{(l)} \times \frac{P(\mathbf{x}_{1:N_t,t}, c_{1:N_t,t}^{(l)} | \theta_{1:K_t^{(l)},t}^{(l)}, m_{1:K_t^{(l)},t}^{(l)})}{P(c_{1:N_t,t}^{(l)} | m_t^{(l)}, \theta_{t-1}^{(l)}, \mathbf{x}_{1:N_t,t})}$ 
39:  end for
40:  for  $l = 1 : L$  do
41:     $w_t^{(l)} \leftarrow \frac{\tilde{w}_t^{(l)}}{\sum_{l=1}^L \tilde{w}_t^{(l)}}$  ▷ normalize weights
42:  end for
43:  Resample particles  $1, \dots, L$  and weights  $w_t^{(1)}, \dots, w_t^{(L)}$  ▷ Section 6.3
44: end for

```

where samples can be drawn from $P(\theta_{k,t}|\theta_{k,t-1}\mathbf{x}_{1:N,t})$ using the Bayesian updates found in (24), (25), (26), and (28), where both the $\mathbf{x}_{1:N,t}$ and auxiliary variables $z_{k,t,1:M}$ are taken to be the observations.

6.3 Resampling Particles and Particle Weights

At each time step $t \in \{1, \dots, T\}$, after all of the L particles have been sampled and their associated weights computed, a resampling step is carried out. In this step, L new particles are sampled from current set of L particles. Resampling strategies such as those described in [28] and [21] can be used.

6.4 From Inference to Tracking Results

Each inferred cluster is taken to be a distinct object, and the sequence of means and covariance matrices for a given cluster are used to determine the position and spatial region, respectively, of a given object over a sequence of time steps. In particular, the mean parameter is taken to be the centroid of an object, and a 2-dimensional oval centered on the mean that contains a specified percentage of the normal distribution mass (where we refer to the specified percentage as the confidence value) is taken to be the spatial region of the object. We report the maximum a posteriori (MAP) sample as our result.

7 Experiments

This section provides details on performance evaluation metrics that have been developed to quantify results in object detection and tracking studies (and adopted in this paper), synthetic video experiments that verify aspects of the developed technique, and benchmark video experiments that demonstrate the performance of this technique in relation to other strategies (including state-of-the-art, object-specific strategies) that have been developed in recent years.

7.1 Performance Evaluation Metrics

Performance evaluation metrics, which provide a standardized way of quantifying the success of a detection and tracking procedure on a given video, have started to become consistently used in the past four years. The metrics presented in [37] and used in [22, 59, 40] have become well established for evaluating the performance of object detection and tracking in videos and have been

adopted by the Video Analysis and Content Extraction (VACE) program and the Classification of Events, Activities, and Relationships (CLEAR) consortium, two large-scale efforts concerned with video tracking and interaction analysis. The two metrics used to quantify the experimental results in this study are known as the Sequence Frame Detection Accuracy (SFDA) and Average Tracking Accuracy (ATA). Details on how these metrics are defined and computed are given in Appendix B.

The above metrics are dependent upon ground-truth data specifying the positions of each object in each frame throughout a video sequence. In the experiments described below, we recorded the synthetic video ground-truth during construction of the videos (described in Section 7.3), and we used the Video Performance Evaluation Resource (ViPER) ground-truth software [20], an open source tool commonly used in the video tracking community, to author ground-truth data for each of the benchmark datasets.

The ground-truth authored by the ViPER tool took the form of bounding boxes denoting the spatial position of each object at each time step. Consequentially, to find the spatial overlap between results and ground-truth, which is intrinsic to both metrics, a rectangular bounding box was needed per object per time step from the results of the algorithm. We took the maximal and minimal axially aligned values of the oval inferred by our algorithm (as described in Section 6.4) to be the sides of a representative bounding box for a given object at a given frame.

7.2 Data Extraction in Experiments

Frame differencing was used in all experiments to identify pixels exhibiting motion. For each pixel $\mathbf{x} = (x_1, x_2, t)$ recorded during frame differencing, we also extracted color information. Specifically, we specified a square, L pixels in length, centered on (x_1, x_2) , that contained a set of pixels surrounding \mathbf{x} in frame t . We chose to capture the hue for each pixel. The set of possible hue values (i.e. the range of hues to which a pixel may be assigned) was partitioned into V bins, and the number of pixels with a color value lying in each of the bins yielded the V dimensional vector of color counts. For all experiments, we chose $V = 10$.

7.3 Synthetic Video Datasets

Each of the following synthetic videos consists of a sequence of 200 images (each of size 500×500 pixels) containing a number of smaller colored squares of different

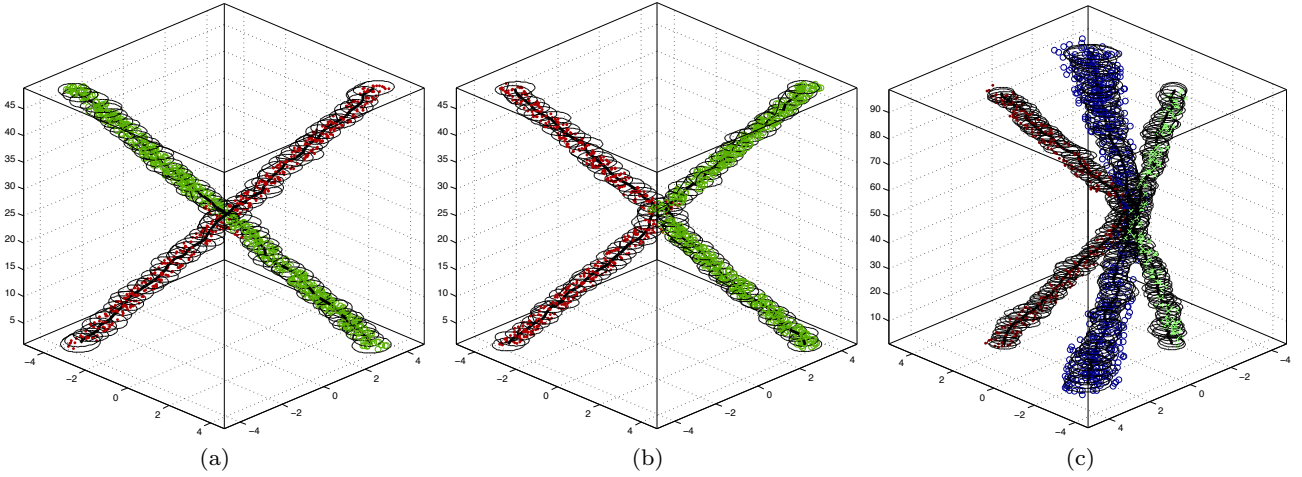


Fig. 4: Each plot shows a sample from the posterior distribution of the model for synthetic experiments one (a and b) and two (c), where the vertical axis represents frame number, the horizontal axes represent spatial position, objects are denoted by marker colors and marker types, and the mean and standard deviation are shown. In all cases, the objects are successfully tracked through occlusion, whether they travel in a straight line (a), reverse direction (b), or do a combination of both (c).

(and potentially time-varying) sizes moving at varied speeds and trajectories over a black background. The synthetic videos contain instances of occlusion (where one or more objects are briefly hidden) and objects with time-varying appearances and behaviors, as these notoriously decrease the accuracy of detection and tracking. After each video was constructed, the extraction procedure described in Section 7.2 (using $L = 3$) and inference procedures described in Section 6 were carried out to return a sequence of multivariate-normal-parameters (means and covariance matrices), which are used to determine a sequence of positions and ovals approximating, respectively, the locations and shapes of a tracked object over each frame that it is present in the video (as outlined in Section 6.4).

The first synthetic video experiment aimed to test the ability of the model and inference procedure to maintain the identity of independent objects based on color information alone. Two videos were constructed, both containing a red square (rgb value $[255, 0, 0]$ and size 20×20 pixels) and a blue square (rgb value $[0, 0, 255]$ and size 20×20 pixels). In both videos, the squares begin at opposite sides of the scene at frame $f = 1$ and travel towards each other, arriving at the same location at $f = 100$ (where the blue square occludes the red square). The second half of the two videos differ in that both squares in the first video continue in the same direction and end at the other's starting position at $f = 200$, and both squares in the second video reverse directions and end at their initial starting positions at $f = 200$. The frame difference extraction yields

identical spatial features in both videos; hence, successful tracking depends fully on the incorporation of color information into the model.

Parameters were set to the same values for inference on both videos: $\alpha = 0.1$, $\rho = 0.3$, $M = 10$, $\mu_0 = (0, 0)$, $\kappa_0 = 0.05$, $\nu_0 = 5$, $\Lambda_0 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, and $\mathbf{q}_0 = (5, \dots, 5)$. Inference was carried out using the MCMC algorithm (Section 6.1); the MAP sample correctly tracked both colored squares through occlusion in both videos, and is shown in Figure 4.

The second synthetic video experiment aimed to test tracking performance under occlusion, object appearance change, and motion change. A video was constructed showing a red square (rgb value $[255, 0, 0]$), a green square (rgb value $[0, 255, 0]$), and a blue square (rgb value $[0, 0, 255]$). The red square was of size 20×20 pixels, the blue square was of size 15×15 pixels, and the green square began at size 50×50 pixels at frame $f = 1$, linearly shrinks to 10×10 pixels at $f = 100$, then linearly grows back to 50×50 pixels by the end of the video, $f = 200$. Furthermore, the red and blue squares display the same behavior as in the second video of the first synthetic experiment (they begin at opposite sides of the scene traveling towards each other, cross at the center of the scene at $f = 100$, and reverse direction, ending at their initial positions at $f = 200$). The green square begins at a point equidistant from the other two squares, intersects with them as they overlap (causing the blue square to occlude the other two), and continues on in a direction at a 20 degree angle from its initial trajectory.

Parameters were set to the same values chosen in the first synthetic experiment. The MCMC inference algorithm correctly tracked all three objects through occlusion and inferred the appearance and size shifts. Figure 4 shows a sample from the posterior distribution of the cluster parameters, where the mean and oval representation of the covariance matrix (with 0.5 confidence value) are overlayed on the data. The data is plotted with time on the vertical axis, and the assignment of each data point to one of the three inferred clusters is denoted by color and marker type.

7.4 Benchmark Video Datasets

Benchmark video datasets for object tracking and detection have been produced to provide standard scenes on which researchers can compare detection and tracking results. These videos have been primarily produced for surveillance-related workshops—notably, for the International Workshop on Performance Evaluation of Tracking and Surveillance (PETS)—which provides researchers with video datasets and algorithmic goals on which to focus. Three commonly used benchmark videos from PETS workshops—one used in PETS2000, one in PETS2001, and one used both in PETS2009 and PETS2010—were chosen to demonstrate the method presented in this study. The performance metrics and benchmark datasets allow the methods developed in this paper to be quantitatively compared against other detection and tracking algorithms.

7.4.1 PETS2000 and PETS2001

The PETS2000 and PETS2001 video datasets both consist of a small number of humans and vehicles traveling across a parking lot, with video taken from above, emulating what might be recorded by standard outdoor surveillance equipment. The “Test Sequence”, a set of images from a monocular, stationary camera, was used from the PETS2000 workshop, and “View Two of Dataset 1”, also taken via a monocular, stationary camera, was used from the PETS2001 workshop. The MCMC algorithm (described in Section 6.1) was used for inference in these experiments.

Due to the computation required for the MCMC batch inference method (discussed further in Section 8), only the final 1000 frames of the video were used from both datasets. Extraction was performed with frame differencing as described in Section 7.2, using $L = 3$. Parameter values were set to the same values as in the synthetic experiments ($\alpha = 0.1, \rho = 0.3, M = 10, \boldsymbol{\mu}_0 = (0, 0), \kappa_0 = 0.05, \nu_0 = 5, \Lambda_0 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, and $\mathbf{q}_0 = (5, \dots, 5)$). The MCMC sampler was successful for both

benchmark videos; each object was detected, tracked, and its shape estimated in manner very consistent with the ground-truth. The results for the PETS2000 dataset are displayed in Figure 5a and for the PETS2001 dataset in Figure 5b; in these figures, a sample from the posterior distribution of the cluster parameters is overlayed on the extracted data over a sequence of frames, where the assignment of each data point is represented by color and marker type.

To calculate performance metrics (both SFDA and ATA), one must specify a confidence value that allows the oval representing the region occupied by an object to be computed from the inferred covariance matrix of each cluster (as discussed in Section 6.4). The performance metrics were found for a range of confidence intervals, and the resulting curves for both the PETS2000 and PETS2001 video are shown in Figure 6.

7.4.2 PETS2009/2010

A video dataset used in both the PETS2009 and PETS2010 conferences, called “S2.L1 at time sequence 12.34” was chosen for experimentation due to its prominence in a number of studies [22, 2, 4, 17, 5, 6, 29, 1, 63]. This dataset consists of a monocular, stationary camera, 794 frame video sequence. The entire video sequence was used in this experiment.

Due to the large number of frames and objects in this video, the SMC algorithm (described in Section 6.2) was used for inference. This method of sequential inference was observed, on this dataset, to converge to a better sample in a shorter period of time in comparison with the MCMC algorithm.

Extraction was performed with frame differencing as described in Section 7.2, using $L = 3$, and parameters for the model were chosen to be $\alpha = 0.1, \rho = 0.8, M = 10, \boldsymbol{\mu}_0 = (0, 0), \kappa_0 = 0.05, \nu_0 = 6, \Lambda_0 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, and $\mathbf{q}_0 = (3, \dots, 3)$. Additionally, as with the other video datasets, ground-truth bounding boxes around each object were authored using the ViPER tool.

The SMC inference algorithm yielded an estimate of the posterior distribution of the model, from which the object detection and tracking results were obtained (as described in Section 6.4). In Figure 7, the MAP sample from the posterior distribution over the cluster parameters is overlayed on the extracted data over a sequence of frames, where the assignment of each data point is represented by color and marker type.

7.4.3 Comparison with Other Methods

In [22], performance metrics (including the SFDA and ATA) were computed for a number of studies that carried out object detection and tracking for the PETS20-

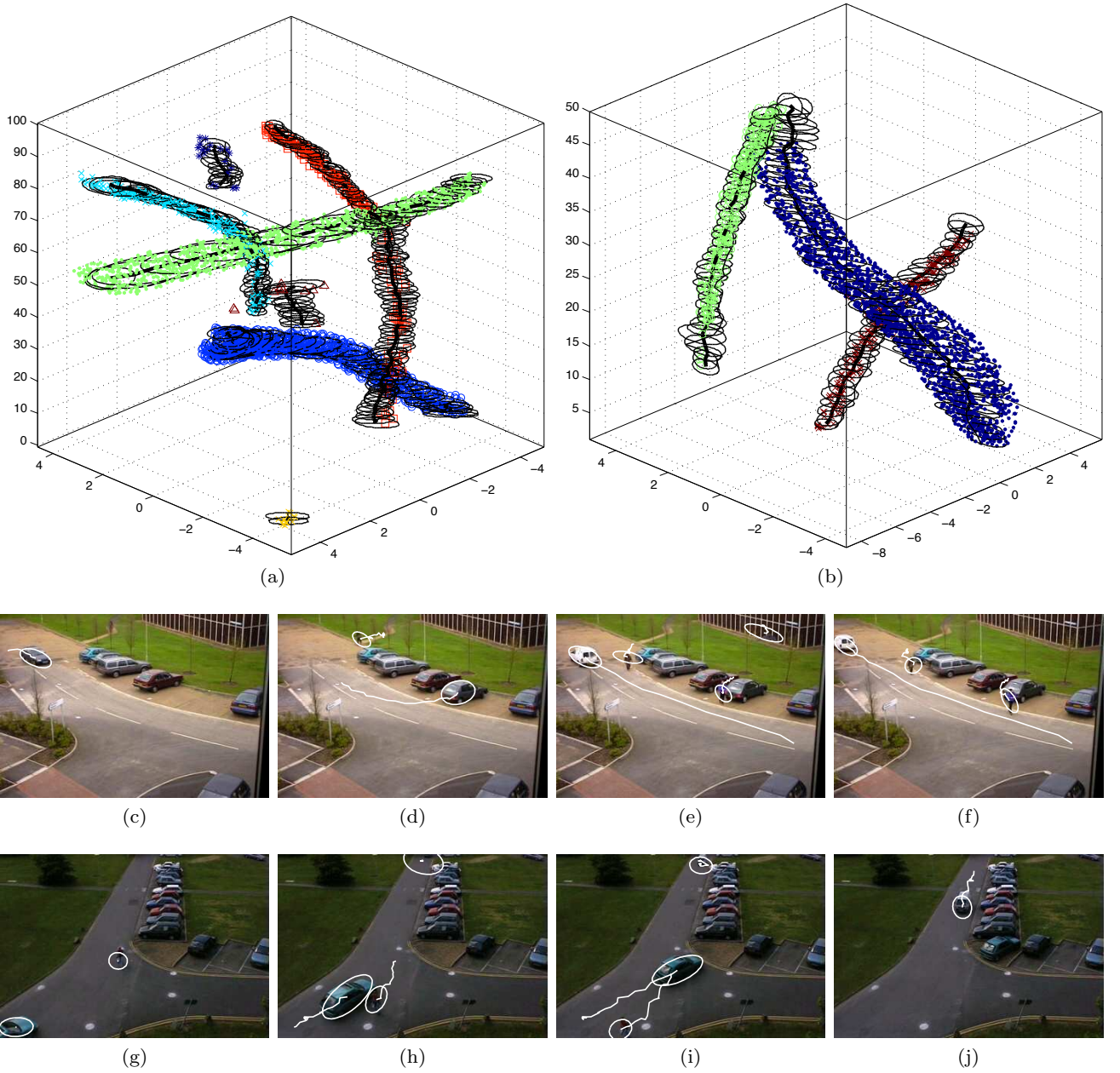


Fig. 5: Results from the PETS2000 (a) and PETS2001 (b) dataset. Both plots show a sample from the posterior distribution of the state, where the vertical axis denotes time, the horizontal axes represent spatial position, color represents assignment, and the mean and standard deviation are shown. Below are four frames from the PETS2000 (c-f) and PETS2001 (g-j) sequence with one posterior sample mean and covariance matrix representation shown for each frame (and one sample mean shown for the previous 20 frames).

09/2010 dataset. As this dataset consists solely of humans, all ten of the algorithms presented for comparison were developed for the specific purpose of people tracking (i.e. not for general detection and tracking of arbitrary objects). As a consequence, many of these studies use externally developed (and trained) state-of-the-art human detectors, exploit the orientation of the humans in this specific dataset, or apply motion

models based on assumptions about human motion. In particular, Breitenstein et al. [6] base their tracking on output from an externally trained human-specific detector; Yang et al. [63] assume they are tracking an upright person, and perform feet and head detection; Conte et al. [17] group foreground fragments based on geometry of the human shape to be recognized and look for shadows often present in human surveillance scenarios;

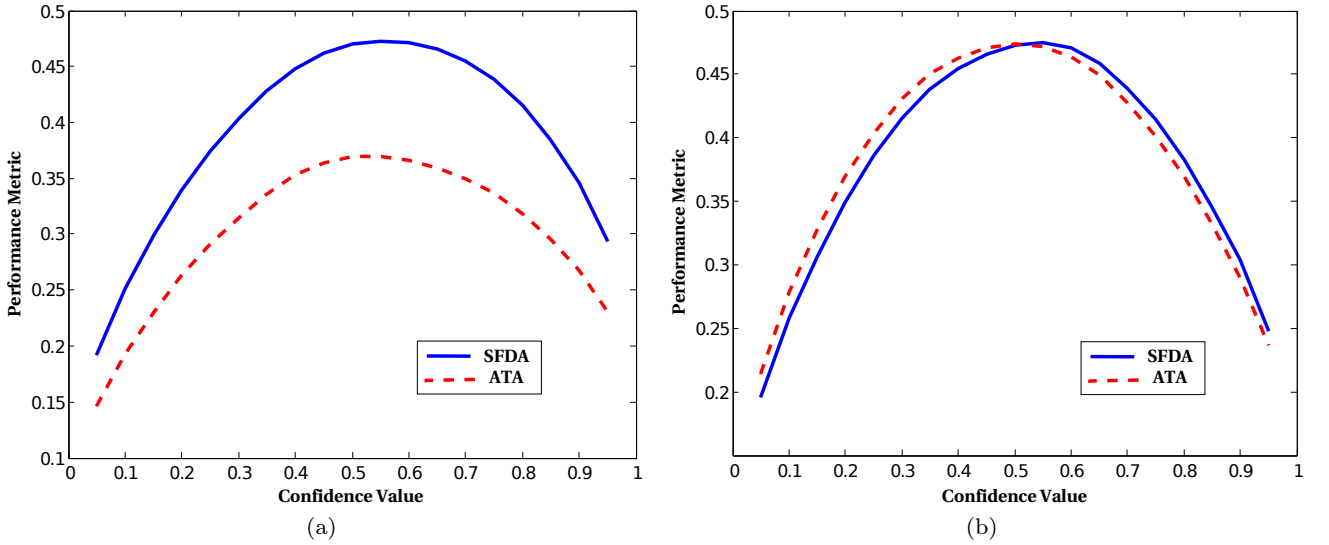


Fig. 6: The SFDA (blue solid line) and ATA (red dashed line) vs confidence values from which an object’s oval region is computed for (a) PETS2000 and (b) PETS2001 video datasets.

Berclaz et al. [4] use an external detector that makes use of multiple camera views and models each human as a cylinder; Alahi et al. [1] base their method on modelling the silhouettes of humans; Bolme et al. [5] train a human specific detector; Ge et al. [29] provide their algorithm with estimates of typical human size and orientation; and Arsic et al. [2] localize human feet positions.

We compare SFDA and ATA results of our strategy with these methods to show that our arbitrary object framework can yield comparable results even when compared with object-specific trackers. Table 1 shows performance metric results for comparison (data published with permission from the authors of [22]). Our method achieves the fourth best SFDA and third best ATA.

7.4.4 Sensitivity Analysis

SMC inference on the PETS2009/2010 video dataset was carried out for a range of the Generalized Polya Urn parameter values, α and ρ . The performance metric measures, SFDA and ATA, were computed for each combination of these two parameters. This sensitivity investigation focused on these parameters due to their potential to have a large effect on object detection accuracies. The α values tested included $\{0.01, 0.1, 1, 10, 100\}$, and the ρ values tested included $\{0.7, 0.75, 0.8, 0.85, 0.9\}$.

The SFDA and ATA achieve their maximal values at different α and ρ parameters, though both achieve a reasonably optimal value at the intermediate parameter values $\alpha = 10$ and $\rho = 0.85$. Detection and tracking

METHOD NAME	SFDA	ATA
Breitenstein [6]	0.57	0.30
Yang [63]	0.55	0.45
Conte [17]	0.53	0.06
GPUDDPM	0.51	0.30
Berclaz [4]	0.48	0.15
Alahi 1 [1]	0.43	0.04
Alahi 2 [1]	0.42	0.05
Bolme 1 [5]	0.41	NA
Ge [29]	0.38	0.04
Bolme 2 [5]	0.34	NA
Arsic [2]	0.18	0.02

Table 1: SFDA and ATA performance metric results are shown for our method (in bold) and for ten other algorithms on the PETS2009/2010 benchmark dataset. Results are listed in descending order of the SFDA value. The results were provided by the authors of [22].

performance was also shown to be fairly robust to minor variations in these parameter values.

8 Conclusion

We have presented a new model for the unsupervised detection and tracking of arbitrary objects in videos. The primary intention of this technique is to reduce the need for detection or localization methods tailored to specific object types and serve as a general framework applicable to videos with varied objects, backgrounds, and film qualities. The GPUDDPM, a time-dependent Dirichlet process mixture, has been introduced, and we have shown how inference on this model

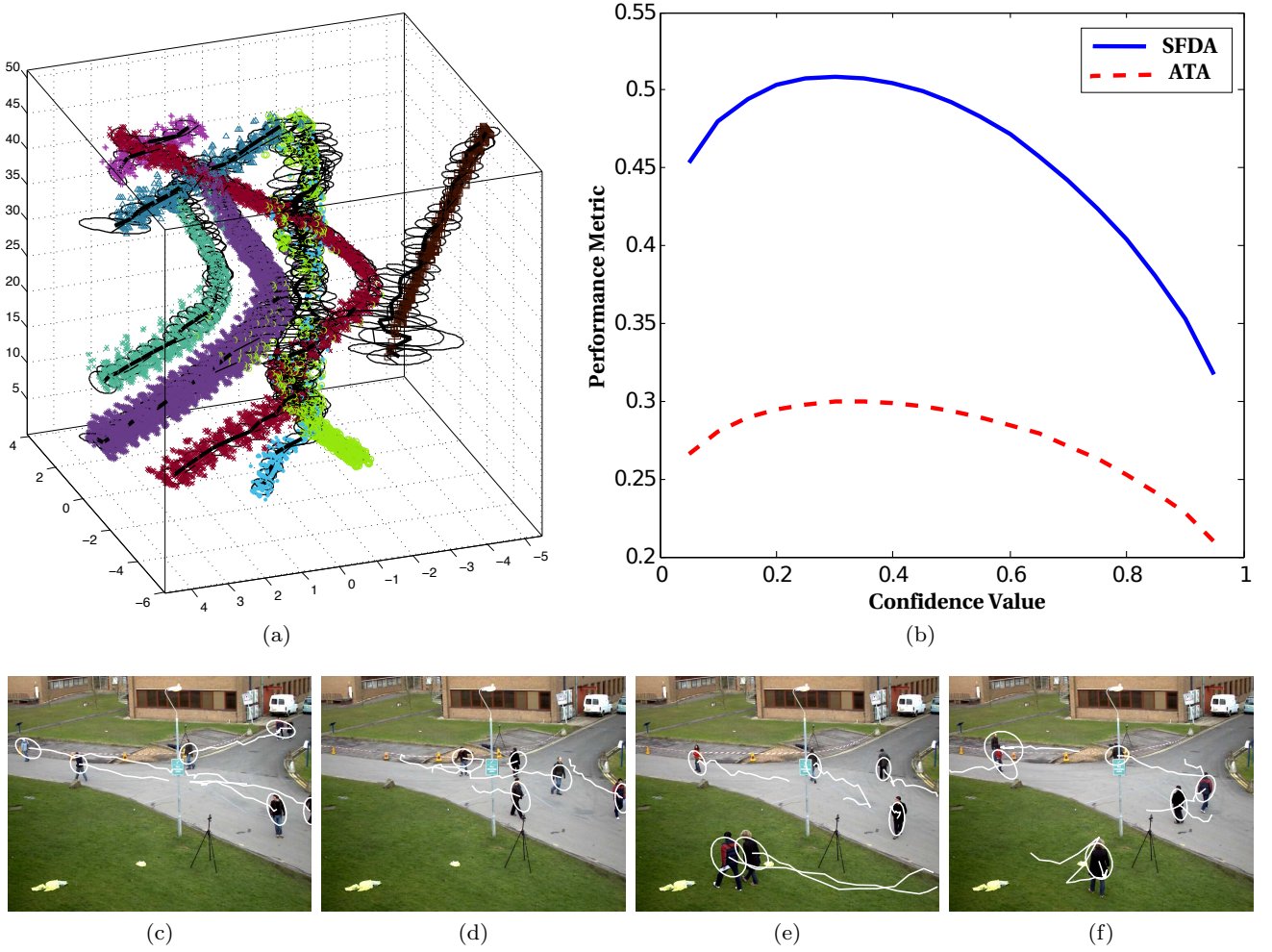


Fig. 7: (a) Results for the PETS2009/2010 dataset, showing a sample from the posterior distribution of the state for frames 1-50, where the vertical axis denotes time, the horizontal axes represent spatial position, color represents assignment, and the mean and standard deviation are shown. (b) performance metrics vs. covariance confidence interval threshold. Below (c-f) are four frames with one posterior sample mean and covariance matrix representation shown for each frame (and one sample mean shown for the previous 20 frames).

allows us to achieve detection and tracking results. Furthermore, we have demonstrated a specific implementation of the model using spatial and color pixel data extracted via frame differencing and provided two algorithms for performing Bayesian inference on the model to accomplish detection and tracking. Both algorithms were carried out on multiple synthetic and benchmark multi-object video datasets in order to demonstrate an ability to accomplish unsupervised detection and tracking of arbitrary objects in both manufactured and real world settings. We have described and computed standard performance metrics for our technique’s detection and tracking results, and found it to be comparable with state-of-the-art object-specific detection and tracking methods designed for people tracking in the PETS2009/2010 video dataset. Results from the syn-

thetic and benchmark video datasets illustrate the ability of the technique described in this paper to, without modification, perform completely unsupervised detection and tracking of objects with diverse physical characteristics moving over non-uniform backgrounds and through occlusion.

References

1. A. Alahi, L. Jacques, Y. Boursier, and P. Vandergheynst. Sparsity-driven people localization algorithm: Evaluation in crowded scenes environments. In *Performance Evaluation of Tracking and Surveillance (PETS-Winter), 2009 Twelfth IEEE International Workshop on*, pages 1–8. IEEE, 2009.
2. D. Arsic, A. Lyutskanov, G. Rigoll, and B. Kwolek. Multi camera person tracking applying a graph-cuts based

- foreground segmentation in a homography framework. In *Performance Evaluation of Tracking and Surveillance (PETS-Winter), 2009 Twelfth IEEE International Workshop on*, pages 1–8. IEEE, 2009.
3. C. Beleznaï, B. Fruhstuck, and H. Bischof. Human tracking by fast mean shift mode seeking. *Journal of Multimedia*, 1(1):1–8, 2006.
 4. J. Berclaz, F. Fleuret, and P. Fua. Multiple object tracking using flow linear programming. In *Performance Evaluation of Tracking and Surveillance (PETS-Winter), 2009 Twelfth IEEE International Workshop on*, pages 1–8. IEEE, 2009.
 5. D.S. Bolme, Y.M. Lui, BA Draper, and JR Beveridge. Simple real-time human detection using a single correlation filter. In *Performance Evaluation of Tracking and Surveillance (PETS-Winter), 2009 Twelfth IEEE International Workshop on*, pages 1–8. IEEE, 2009.
 6. M.D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool. Markovian tracking-by-detection from a single, uncalibrated camera. *Work*, 2009.
 7. Michael D. Breitenstein, Fabian Reichlin, Bastian Leibe, Esther Koller-Meier, and Luc Van Gool. Robust tracking-by-detection using a detector confidence particle filter. In *IEEE International Conference on Computer Vision*, October 2009.
 8. G.J. Brostow and R. Cipolla. Unsupervised Bayesian detection of independent motion in crowds. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 1, pages 594–601. IEEE, 2006.
 9. T. Brox and J. Malik. Object segmentation by long term analysis of point trajectories. *Computer Vision–ECCV 2010*, pages 282–295, 2010.
 10. T. Brox, M. Rousson, R. Deriche, and J. Weickert. Unsupervised segmentation incorporating colour, texture, and motion. In *Computer Analysis of Images and Patterns*, pages 353–360. Springer, 2003.
 11. F. Caron, M. Davy, and A. Doucet. Generalized Polya urn for time-varying Dirichlet process mixtures. In *23rd Conference on Uncertainty in Artificial Intelligence (UAI'2007), Vancouver, Canada, July 2007*, 2007.
 12. S.Y. Chien, S.Y. Ma, and L.G. Chen. Efficient moving object segmentation algorithm using background registration technique. *Circuits and Systems for Video Technology, IEEE Transactions on*, 12(7):577–586, 2002.
 13. Kiam Choo and D.J. Fleet. People tracking using hybrid monte carlo filtering. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 2, pages 321–328 vol.2, 2001.
 14. H. Chu, S. Ye, Q. Guo, and X. Liu. Object tracking algorithm based on camshift algorithm combining with difference in frame. In *Automation and Logistics, 2007 IEEE International Conference on*, pages 51–55. IEEE, 2007.
 15. R.T. Collins. Mean-shift blob tracking through scale space. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 2, pages II–234. IEEE, 2003.
 16. D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(5):564–577, may 2003.
 17. D. Conte, P. Foggia, G. Percannella, and M. Vento. Performance evaluation of a people tracking system on pets2009 database. In *Advanced Video and Signal Based Surveillance (AVSS), 2010 Seventh IEEE International Conference on*, pages 119–126. IEEE, 2010.
 18. R. Cucchiara, C. Grana, G. Tardini, and R. Vezzani. Probabilistic people tracking for occlusion handling. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 1, pages 132–135. IEEE, 2004.
 19. S.L. Dockstader and A.M. Tekalp. Multiple camera tracking of interacting and occluded human motion. *Proceedings of the IEEE*, 89(10):1441–1455, 2001.
 20. D. Doermann and D. Mihalcik. Tools and techniques for video performance evaluation. In *Pattern Recognition, 2000. Proceedings. 15th International Conference on*, volume 4, pages 167–170. IEEE, 2000.
 21. R. Douc and O. Cappé. Comparison of resampling schemes for particle filtering. In *Image and Signal Processing and Analysis, 2005. ISPA 2005. Proceedings of the 4th International Symposium on*, pages 64–69. IEEE, 2005.
 22. A. Ellis and J. Ferryman. Pets2010 and pets2009 evaluation of results using individual ground truthed single views. In *Advanced Video and Signal Based Surveillance (AVSS), 2010 Seventh IEEE International Conference on*, pages 135–142. IEEE, 2010.
 23. L. Fei-Fei and P. Perona. A Bayesian hierarchical model for learning natural scene categories. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 524–531. IEEE, 2005.
 24. T.S. Ferguson. A Bayesian analysis of some nonparametric problems. *The annals of statistics*, pages 209–230, 1973.
 25. K. Fragkiadaki and J. Shi. Detection free tracking: Exploiting motion and topology for segmenting and tracking under entanglement. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 2073–2080. IEEE, 2011.
 26. A.R. Francois. Real-time multi-resolution blob tracking. Technical report, DTIC Document, 2004.
 27. J. Gasthaus, F. Wood, D. Görür, and Y. W. Teh. Dependent Dirichlet process spike sorting. In *Advances in Neural Informations Processing Systems 22*, 2008.
 28. Jan Gasthaus. Spike sorting using time-varying Dirichlet process mixture models, 2008.
 29. W. Ge and R.T. Collins. Evaluation of sampling-based pedestrian detection for crowd counting. In *Performance Evaluation of Tracking and Surveillance (PETS-Winter), 2009 Twelfth IEEE International Workshop on*, pages 1–7. IEEE, 2009.
 30. A. Gelman. *Bayesian data analysis*. CRC press, 2004.
 31. J.E. Griffin and M.F.J. Steel. Order-based dependent Dirichlet processes. *Journal of the American statistical Association*, 101(473):179–194, 2006.
 32. M. Han, A. Sethi, W. Hua, and Y. Gong. A detection-based multiple object tracking method. In *Image Processing, 2004. ICIP'04. 2004 International Conference on*, volume 5, pages 3065–3068. IEEE, 2004.
 33. W.D. Hong, T.H. Lee, and P.C. Chang. Real-time foreground segmentation for the moving camera based on h. 264 video coding information. In *Future Generation Communication and Networking (FGCN 2007)*, volume 1, pages 385–390. IEEE, 2007.
 34. M. Isard and J. MacCormick. Bramble: A Bayesian multiple-blob tracker. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 2, pages 34–41. IEEE, 2001.
 35. A.K. Jain, N.K. Ratha, and S. Lakshmanan. Object detection using Gabor filters. *Pattern Recognition*, 30(2):295–309, 1997.
 36. A.D. Jepson, D.J. Fleet, and T.F. El-Maraghi. Robust online appearance models for visual tracking. *IEEE*

- Transactions on Pattern Analysis and Machine Intelligence*, pages 1296–1311, 2003.
37. R. Kasturi, D. Goldgof, P. Soundararajan, V. Manohar, J. Garofolo, R. Bowers, M. Boonstra, V. Korzhova, and J. Zhang. Framework for performance evaluation of face, text, and vehicle detection and tracking in video: Data, metrics, and protocol. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 319–336, 2008.
 38. Z. Khan, T. Balch, and F. Dellaert. An MCMC-based particle filter for tracking multiple interacting targets. *Computer Vision-ECCV 2004*, pages 279–290, 2004.
 39. C. Kim and J.N. Hwang. Fast and automatic video object segmentation and tracking for content-based applications. *Circuits and Systems for Video Technology, IEEE Transactions on*, 12(2):122–129, 2002.
 40. S. Lee and R. Nevatia. Speed performance improvement of vehicle blob tracking system. *Multimodal Technologies for Perception of Humans*, pages 197–202, 2009.
 41. B. Leibe, K. Schindler, N. Cornelis, and L. Van Gool. Coupled object detection and tracking from static cameras and moving vehicles. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(10):1683–1698, Oct. 2008.
 42. J. MacCormick and A. Blake. A probabilistic exclusion principle for tracking multiple objects. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 1, pages 572–578. IEEE, 1999.
 43. S.J. McKenna, Y. Raja, and S. Gong. Tracking colour objects using adaptive mixture models. *Image and Vision Computing*, 17(3-4):225–231, 1999.
 44. S.J. McKenna, S. Jabri, Z. Duric, and H. Wechsler. Tracking interacting people. In *Automatic Face and Gesture Recognition, 2000. Proceedings. Fourth IEEE International Conference on*, pages 348–353. IEEE, 2000.
 45. J. Munkres. Algorithms for the assignment and transportation problems. *Journal of the Society for Industrial and Applied Mathematics*, pages 32–38, 1957.
 46. M. Nedrich and J. Davis. Learning scene entries and exits using coherent motion regions. *Advances in Visual Computing*, pages 120–131, 2010.
 47. Kenji Okuma, Ali Taleghani, Nando De Freitas, O De Freitas, James J. Little, and David G. Lowe. A boosted particle filter: Multitarget detection and tracking. In *ECCV*, pages 28–39, 2004.
 48. N. Paragios and R. Deriche. Geodesic active contours and level sets for the detection and tracking of moving objects. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(3):266–280, Mar 2000.
 49. A.E.C. Pece. Generative-model-based tracking by cluster analysis of image differences. *Robotics and Autonomous Systems*, 39(3-4):181–194, 2002.
 50. P. Pérez, C. Hue, J. Vermaak, and M. Gangnet. Color-based probabilistic tracking. *Computer Vision-ECCV 2002*, pages 661–675, 2002.
 51. M. Piccardi. Background subtraction techniques: a review. In *Systems, Man and Cybernetics, 2004 IEEE International Conference on*, volume 4, pages 3099–3104. IEEE, 2004.
 52. Y. Raja, S.J. McKenna, and S. Gong. Tracking and segmenting people in varying lighting conditions using colour. In *Automatic Face and Gesture Recognition, 1998. Proceedings. Third IEEE International Conference on*, pages 228–233. IEEE, 1998.
 53. V. Reilly, H. Idrees, and M. Shah. Detection and tracking of large number of targets in wide area surveillance. *Computer Vision-ECCV 2010*, pages 186–199, 2010.
 54. A. Senior, A. Hampapur, Y.L. Tian, L. Brown, S. Pankanti, and R. Bolle. Appearance models for occlusion handling. *Image and Vision Computing*, 24(11):1233–1243, 2006.
 55. S. Sista and R.L. Kashyap. Unsupervised video segmentation and object tracking. *Computers in Industry*, 42(2-3):127–146, 2000.
 56. J. Sivic, B.C. Russell, A.A. Efros, A. Zisserman, and W.T. Freeman. Discovering objects and their location in images. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, pages 370–377. IEEE, 2005.
 57. X. Song and R. Nevatia. A model-based vehicle segmentation method for tracking. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pages 1124–1131. IEEE, 2005.
 58. C. Stauffer. Estimating tracking sources and sinks. In *Computer Vision and Pattern Recognition Workshop, 2003. CVPRW'03. Conference on*, volume 4, pages 35–35. IEEE, 2003.
 59. M. Taj, E. Maggio, and A. Cavallaro. Multi-feature graph-based object tracking. *Multimodal Technologies for Perception of Humans*, pages 190–199, 2007.
 60. J. Vermaak, A. Doucet, and P. Pérez. Maintaining multimodality through mixture tracking. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 1110–1116. IEEE, 2003.
 61. D. Wang. Unsupervised video segmentation based on watersheds and temporal tracking. *Circuits and Systems for Video Technology, IEEE Transactions on*, 8(5):539–546, 1998.
 62. Frank Wood and Michael J. Black. A nonparametric bayesian alternative to spike sorting. *Journal of Neuroscience Methods*, 173(1):1–12, 2008.
 63. J. Yang, PA Vela, Z. Shi, and J. Teizer. Probabilistic multiple people tracking through complex situations. In *11th IEEE International Workshop on Performance Evaluation of Tracking and Surveillance*, 2009.
 64. D. Zhang and G. Lu. Segmentation of moving objects in image sequence: A review. *Circuits, systems, and signal processing*, 20(2):143–183, 2001.
 65. G. Zhang, J. Jia, W. Xiong, T.T. Wong, P.A. Heng, and H. Bao. Moving object extraction with a hand-held camera. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE, 2007.
 66. Y. Zhou and H. Tao. A background layer model for object tracking through occlusion. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 1079–1085. IEEE, 2003.
 67. Guangyu Zhu, Changsheng Xu, Qingming Huang, and Wen Gao. Automatic multi-player detection and tracking in broadcast sports video using support vector machine and particle filter. In *Multimedia and Expo, 2006 IEEE International Conference on*, pages 1629–1632, july 2006.

A - Appendix: Model Background

We give background on mixture models, Bayesian mixture models, the Dirichlet process, Dirichlet process (infinite) mixture models, and dependent Dirichlet process mixture models.

A.1 Finite Mixture Model

A finite mixture model can be thought of as a probability distribution for an observation x_i formulated as a linear combination of K mixture components (which we also refer to as ‘clusters’), where each mixture component is a probability distribution for x_i with some parametric form, and the coefficients of the linear combination sum to one. The finite mixture model can be written as

$$P(x_i) = \sum_{k=1}^K P(c_i = k)P(x_i|\theta_k) \quad (37)$$

$\forall i \in \{1, \dots, N\}$, where $c_i \in \{1, \dots, K\}$ denotes the assignment of x_i to a given mixture and θ_k denotes the parameters of the k^{th} mixture component. Note that by choosing $P(c_i = k)$ as coefficients of the linear combination, it is ensured that these coefficients sum to one. We also define $p_k := P(c_i = k)$ for $k \in \{1, \dots, K\}$. We can therefore write this model generatively as

$$\begin{aligned} c_i | p_1, \dots, p_K &\sim \text{Discrete}(p_1, \dots, p_K) \\ x_i | c_i, \theta_{c_i} &\sim F(\theta_{c_i}) \end{aligned} \quad (38)$$

$\forall i \in \{1, \dots, N\}$, where the x_i are observations, the c_i are the mixture component assignments associated with each observation, the θ_{c_i} are parameters defining the c_i^{th} mixture component (i.e. the distribution to be mixed, $F(\theta_{c_i})$), and the “Discrete” distribution refers to a multinomial distribution whose parameters are a 1-of- K vector (i.e. a vector of counts that sums to one).

A.2 Bayesian (Finite) Mixture Model

The finite mixture model of Section A.1 can be extended to a Bayesian mixture model by viewing parameters that were previously point values, θ_{c_i} (the mixture component parameters) and p_1, \dots, p_K (the mixture component assignment weights), as random variables and providing each with a prior distribution. In this case, the prior distribution \mathbb{G}_0 is placed on the mixture component parameters, and the prior distribution $\text{Dir}(\alpha/K, \dots, \alpha/K)$ is placed on the mixture component assignment weights. The resulting Bayesian mixture model can be formulated generatively as

$$\begin{aligned} p_1, \dots, p_K &\sim \text{Dir}(\alpha/K, \dots, \alpha/K) \\ \theta_1, \dots, \theta_K &\sim \mathbb{G}_0 \\ c_i | p_1, \dots, p_K &\sim \text{Discrete}(p_1, \dots, p_K) \\ x_i | c_i, \theta_{c_i} &\sim F(\theta_{c_i}) \end{aligned} \quad (39)$$

$\forall i \in \{1, \dots, N\}$, where the x_i are observations, the c_i are the mixture component assignments associated with each observation, the θ_k are parameters defining the k^{th} mixture component (i.e. the distribution to be mixed, $F(\theta_k)$), the θ_k are drawn from a prior distribution \mathbb{G}_0 , and p_1, \dots, p_K are drawn from a Dirichlet prior parameterized by $\alpha/K, \dots, \alpha/K$.

A.3 Dirichlet Process

The Dirichlet process (DP), first introduced by [24] in 1973, may be intuitively viewed as a probability distribution over discrete probability distributions. Accordingly, draws from a DP are probability mass functions (PMFs). A DP is parameterized by a base distribution \mathbb{G}_0 , which is a probability distribution over a set Θ , and a concentration parameter $\alpha \in \mathbb{R}_+$. We say that G is a random PMF distributed according to a DP, written $G \sim \text{DP}(\alpha, \mathbb{G}_0)$, if the following holds for all finite partitions A_1, \dots, A_p of Θ :

$$(G(A_1), \dots, G(A_p)) \sim \text{Dir}(\alpha\mathbb{G}_0(A_1), \dots, \alpha\mathbb{G}_0(A_p)) \quad (40)$$

Where ‘Dir’ denotes a Dirichlet distribution. The parameters \mathbb{G}_0 and α may be intuitively viewed as the mean and precision of the DP. This is due to the fact that if the base distribution \mathbb{G}_0 is a distribution over Θ , $A \subset \Theta$, and $G \sim \text{DP}(\alpha, \mathbb{G}_0)$, then the following holds:

$$\mathbb{E}[G(A)] = \mathbb{G}_0(A) \quad (41)$$

$$\text{Var}[G(A)] = \mathbb{G}_0(A)(1 - \mathbb{G}_0(A))/(\alpha + 1) \quad (42)$$

Hence, the expectation of $G(A)$ is \mathbb{G}_0 , the variance of $G(A) \rightarrow 0$ as $\alpha \rightarrow \infty$, and G converges pointwise to \mathbb{G}_0 when α is unbounded.

A.4 Dirichlet Process (Infinite) Mixture Model

A DPM model, also referred to as an infinite mixture model, is an extension of the Bayesian mixture model described in Section A.2. When using a DP as a prior in a Bayesian mixture model, Θ represents the set of parameters of the component mixture distributions. A DPM may be viewed as allowing the prior distribution over the mixture component parameters in a standard mixture model to be distributed according to a DP; this allows for modeling data where the true number of latent mixture components is unknown and arbitrarily large by letting the number of components remain unbounded (note that only a finite number of these components are assigned to the data). In particular, the DPM can be defined generatively as

$$\begin{aligned} \mathbb{G} | \alpha, \mathbb{G}_0 &\sim \text{DP}(\alpha, \mathbb{G}_0) \\ \phi_i | \mathbb{G} &\sim \mathbb{G} \\ x_i | \phi_i &\sim F(\phi_i) \end{aligned} \quad (43)$$

$\forall i \in \{1, \dots, N\}$, where the x_i are observations, the ϕ_i are parameters defining the mixture component from which the i^{th} observation is drawn (i.e. the distribution to be mixed, $F(\phi_i)$), and the ϕ_i are drawn from a prior distribution \mathbb{G} , which is in turn drawn from a DP with base distribution \mathbb{G}_0 and parameter α . See [27] and [28] for more details on this formulation. Note the difference between the indexing of the clusters in this model and the indexing in the previous two models. This formulation can be shown to be equivalent to the Bayesian mixture model defined in (39), when K is taken to be unbounded; as a result, this model is sometimes called an infinite mixture model. If we let K be the number of distinct mixture components assigned to observations using the above model, we can write the mixture components as $\theta_1, \dots, \theta_K$. We also let c_1, \dots, c_N (where $c_i \in \{1, \dots, K\}$) be class assignment variables that indicate the cluster to which observation x_i is assigned.

A.5 Dependent Dirichlet Process Mixture Model

The goal of DDPM models is to allow modeling of data that is not independent and identically distributed but instead has some underlying dependencies. For example, data generated during video extraction procedures have some associated temporal dependencies, since there exist similarities between features (such as those that encode the spatial positions or appearances of objects) of data at nearby time steps.

To account for the dependent behavior of data, there has been research into models involving a sequence of DPMs, where components of the mixtures are dependent upon (or are sometimes said to be “tied to”) corresponding components at neighboring positions in the sequence. For example, if the data shows temporal dependence, the goal might be to create a sequence of DPMs, one for each time-step, where the components of the mixture at each step are dependent upon corresponding components in the both the following and previous time steps.

More rigorously, we take the definition of a DDPM to be a stochastic process defined on the space of probability distributions over a domain, which are indexed by time, space, or a selection of other covariates in such a way that the marginal distribution at any point in the domain follows a Dirichlet process (adapted from definitions found in [28] and [31]). Hence, a time-dependent DDPM is a model which remains a Dirichlet process, marginally, at each time step, yet allows cluster parameters at a given time step to vary from (and remain dependent upon) the parameters in neighboring time steps.

B - Appendix: Performance Metric Details

This section provides details on the definition and calculation of the performance evaluation metrics, SFDA and ATA, used to quantify detection and tracking results in this study.

B.1 Mapping Ground-Truth to Output

The problem of finding a mapping between a video’s ground-truth tracks and an algorithm’s output tracks is nontrivial, though necessary to solve, in order to compute the performance evaluation metrics used in this study. In short, the typical solution to this problem involves first specifying a performance metric and then choosing the mapping from ground-truth tracks to output tracks which yields the most favorable performance metric value. This process is described in detail by Kasturi et al. [37]; we follow the method outlined in this paper to find an optimal mapping. Similiar to descriptions in [37], we implement the Hungarian algorithm [45] as a polynomial-time ($O(n^3)$) solution to the problem of optimally mapping two sets of tracks once the similarity between any two tracks given some specified metric is established. Additionally, the method employed in [37] allows erroneous and undetected tracks to be left unmapped, which is both desired and necessary in the case where there is a different number of ground-truth and output tracks. Note that once a mapping from a collection of ground-truth tracks to a collection of result tracks has been established, one can determine which result tracks are false positives (the result tracks to which no ground-truth track is assigned) and which ground-truth tracks are true negatives (the ground-truth tracks that are

not assigned to a result track). The numbers of tracks displaying both of these failures are factors in the performance metrics used in this study.

B.2 SFDA and ATA

The two metrics used to quantify performance in this study are known as the the Sequence Frame Detection Accuracy (SFDA) and the Average Tracking Accuracy (ATA). These metrics were developed during VACE Phase II to provide a single, comprehensive metric to describe detection, and one to describe tracking. The following are used in the definitions of the performance metrics:

- G_i denotes the spatiotemporal region occupied by the i th ground-truth object in a video, and $G_i^{(t)}$ denotes the region occupied by the i th ground-truth object in frame t .
- D_i denotes the spatiotemporal region occupied by the i th detected object in a video, and $D_i^{(t)}$ denotes the region occupied by the i th detected object in frame t .
- N_G denotes the total number of unique ground-truth objects in a video, and $N_G^{(t)}$ denotes the number of unique ground-truth objects present at frame t .
- N_D denotes the total number of unique detected objects in a video, and $N_D^{(t)}$ denotes the number of unique detected objects present at frame t .
- N_{frames} denotes the total number of frames in a video, and $N_{\text{frames}}^{(i)}$ denotes the number of frames in which an object i (which can be a ground-truth or detected object, depending on the context) is present in a video.
- N_{mapped} denotes the number of mapped ground-truth/detect pairs in a video, and $N_{\text{mapped}}^{(t)}$ denotes the number of mapped ground-truth/detect pairs present at frame t .

The SFDA metric quantifies the performance of an object detection algorithm as a function of the number of correct detects, false positive detects, missed (true negative) detects, and spatial alignment of detects relative to the ground-truth. The SFDA is calculated by computing the Frame Detection Accuracy at frame t ($FDA^{(t)}$) for each frame in a video sequence. The FDA provides a measure of the alignment between ground-truth and detected objects in a given frame via the overlap ratio of a ground-truth/detect pair, defined to be the ratio of the intersection of ground-truth and detect regions to the union of ground-truth and detect regions. Formally, we can write

$$FDA^{(t)} = \frac{\text{Overlap Ratio}}{\left(\frac{N_G^{(t)} + N_D^{(t)}}{2} \right)} \quad (44)$$

where

$$\text{Overlap Ratio} = \sum_{i=1}^{N_{\text{mapped}}^{(t)}} \frac{|G_i^{(t)} \cap D_i^{(t)}|}{|G_i^{(t)} \cup D_i^{(t)}|} \quad (45)$$

The term $N_{\text{mapped}}^{(t)}$ refers to an optimal mapping between ground-truth and detects at frame t as specified in section B.1 using the $FDA^{(t)}$ as the relevant metric. Given the $FDA^{(t)}$ at each frame, the SFDA can be computed; this metric may

be viewed as the average FDA over all frames of a video sequence. We define

$$\text{SFDA} = \frac{\sum_{t=1}^{N_{\text{frames}}} \text{FDA}^{(t)}}{\sum_{t=1}^{N_{\text{frames}}} \exists \left(N_G^{(t)} \vee N_D^{(t)} \right)} \quad (46)$$

where $\exists \left(N_G^{(t)} \vee N_D^{(t)} \right)$ yields a 1 if either a detected or ground-truth object is present in frame t and a 0 otherwise.

The ATA metric quantifies the performance of an object tracking algorithm as a function of the spatial overlap of a mapped set of sequences of detected object positions to a set of sequences of groundtruth object positions. The ATA is calculated by first computing the Sequence Track Detection Accuracy (STDA), which can be viewed as a tracking performance measure unnormalized in terms of the number of objects. We can write the STDA as

$$\text{STDA} = \sum_{i=1}^{N_{\text{mapped}}} \frac{\sum_{t=1}^{N_{\text{frames}}} \left(\frac{|G_i^{(t)} \cap D_i^{(t)}|}{|G_i^{(t)} \cup D_i^{(t)}|} \right)}{N_{(G_i \cup D_i \neq \emptyset)}} \quad (47)$$

where N_{mapped} refers to an optimal mapping between ground-truth and detected objects as specified in section B.1 using the STDA as the relevant metric, and $N_{(G_i \cup D_i \neq \emptyset)}$ denotes the number of frames in which a given tracked object, the ground truth object to which it is mapped, or both, are present.

Given the STDA for a video sequence, the ATA can be computed by the formula

$$\text{ATA} = \frac{\text{STDA}}{\left(\frac{N_G + N_D}{2} \right)} \quad (48)$$